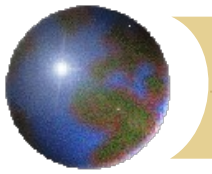


ADVANCED PROGRAMMING 8086 INSTRUCTION SET

UNCLASSIFIED

CF351L02-

01



RISK ASSESSMENT:

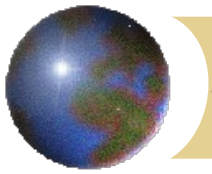
LOW
SAFETY

CONSIDERATIONS:
ELECTRICAL SHOCK HAZARD

ELECTROSTATIC DISCHARGE

ENVIRONMENTAL

CONSIDERATIONS: None



TERMINAL LEARNING *OBJECTIVE* **ACTION:**

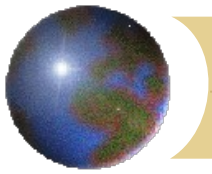
Construct a computer program.

CONDITIONS:

Given a classroom environment, a NIDA 500 trainer, the 8086 instruction set and official OMEMS reference material.

STANDARDS:

Correctly answer 7 of 10 questions on the construction of operation codes.



ENABLING LEARNING

OBJECTIVE

ACTION:

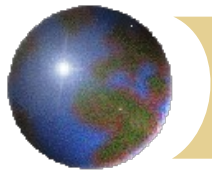
**Identify an op code instruction bit
by bit**

CONDITIONS:

**Given a classroom environment, a NIDA
500 trainer, the 8086 instruction set and
official OMEMS reference material.**

STANDARDS:

**Correctly answer 7 of 10 questions on
the construction of operation
codes.**



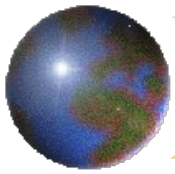
INSTRUCTION OP CODE

Opcode can consist of 8 or 16 bits.

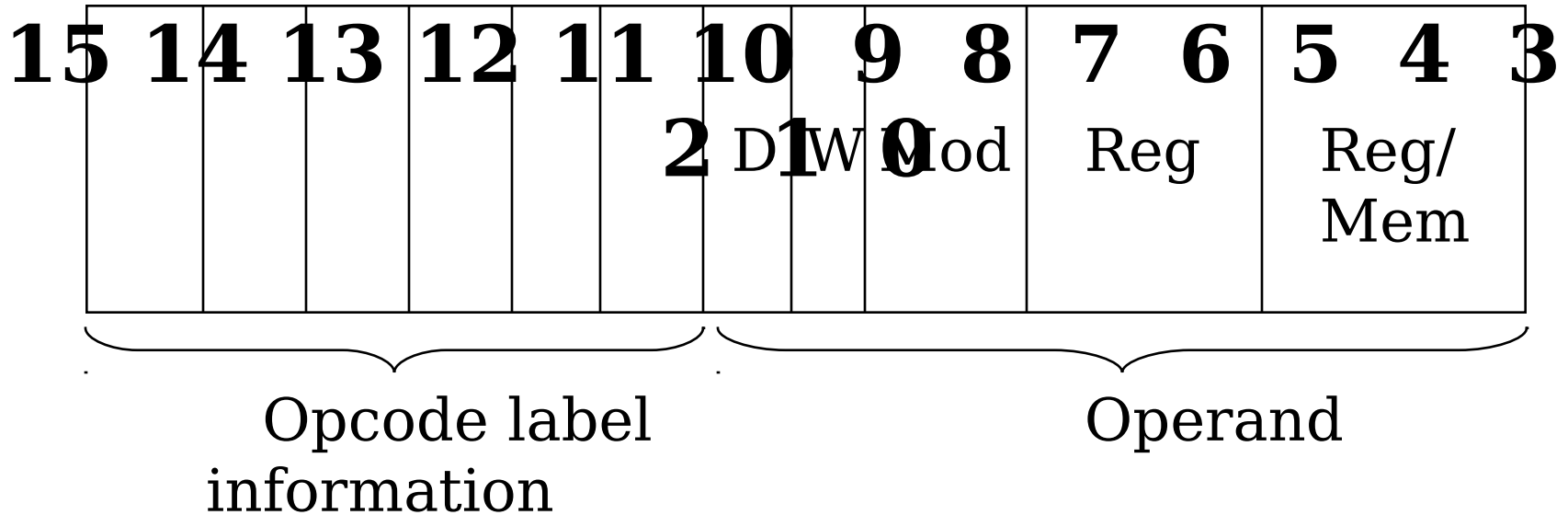
Codes are contained in the code segment.

The first few bits are the “label” of the instruction.

The other bits contain information for determining the effective address of the operand.

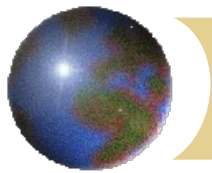


OPCODE BITS



The Opcode label is already predetermined. Only the operand information can be manipulated.

See chart on page A-15



OPCODE EXAMPLES

ADD mem,
reg **Page A-17**

15	14	13	12	11	10	9	8	7	6	5	4
3 0	2 1 0	0	0	0	0	0	x	x	x	x	x
x x x			D W Mod			Reg			Reg/ Mem		

MOV reg,
reg **Page A-15**

1	0	0	0	1	0	1	x	1	1	x	x	x
x x x			D W Mod			Reg			Reg/ Mem			

OUT imm8,
acc **Page A-16**

1	1	1	0	1	1	x	Port selection 70 or 71.
W							

x = fields determined by the
programmer.



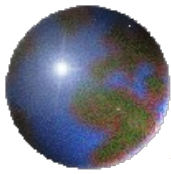
1

1 = Data flow is from R/M field to the REG field.

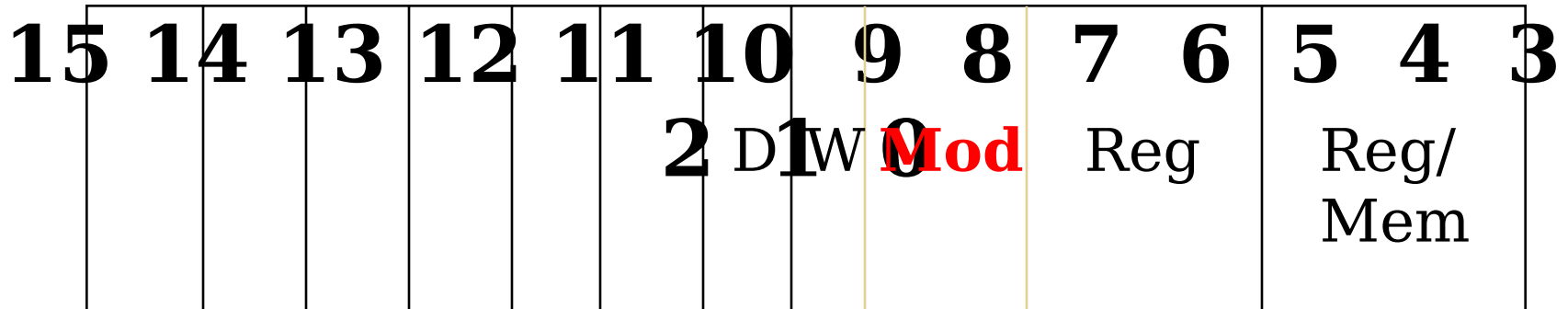
1 = Data flow is from R/M field to the REG field.



0 = BYTE
1 = WORD



OPCODE BITS



BITS 7,6 MOD FIELD

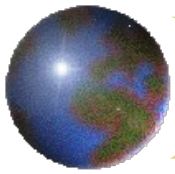
SELECTS THE ADDRESSING MODE FOR THE
SELECTED
INSTRUCTION.

00 = DATA MEM ADDRESSING MODE (No displacement)
Except when the R/M field value equals 110.

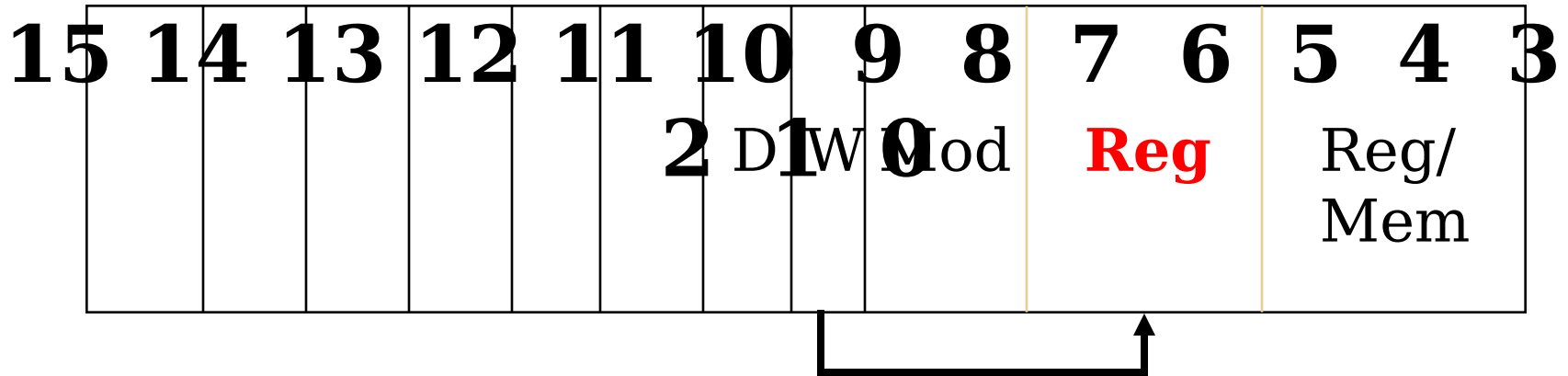
01 = DATA MEM ADDRESSING MODE (8 Bit sign displacement)

10 = DATA MEM ADDRESSING MODE (16 Bit sign displacement)

11 = REGISTER ADDRESSING MODE



OPCODE BITS



BITS 5, 4, 3 REG BIT

determines the register assignment. The word bit is used to determine the value of this field.

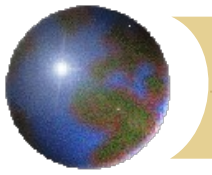
See Register Field Table Page A-14



etermines the memory addressing mode when the MOD field is 00, 01, 10 or the register assignment when the MOD is 11

See Register Field Table Page
A-14

CF351L02-
12



MEMORY ADDRESSING MODES

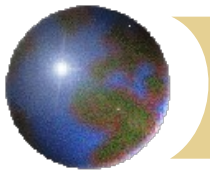
MOD - MODE FIELD

00 = No displacement (Except when
the R/M field equals 110)

01 = 8 bit displacement follows

10 = 16 bit displacement follows

**11 = mem specifies reg instead of
address mode**



MEMORY ADDRESSING MODES

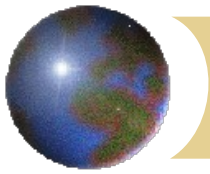
Determines the addressing mode (memory or register) depending on the value of the MOD field.

A three bit value that is determined by the memory addressing mode selected by the programmer.

R/M	MOD = 00	MOD = 01	MOD = 10	MOD = 11
000	BX + SI	BX + SI + 8	BX + SI + 16	W=0 SI + 16
001	AL AX BX + DI	BX + DI + 8	BX + DI + 16	
010	CL CX BP + SI	BP + SI + 8	BP + SI + 16	
011	DL DX BP + DI	BP + DI + 8	BP + DI + 16	
100	BL BX SI	SI + 8	SI + 16	
101	AH SP DI	DI + 8	DI + 16	
110	CH BP DISP 16	BP + 8	BP + 16	
111	DH SI BX	BX + 8	BX + 16	
	BH DI			

Indirect
Transfer

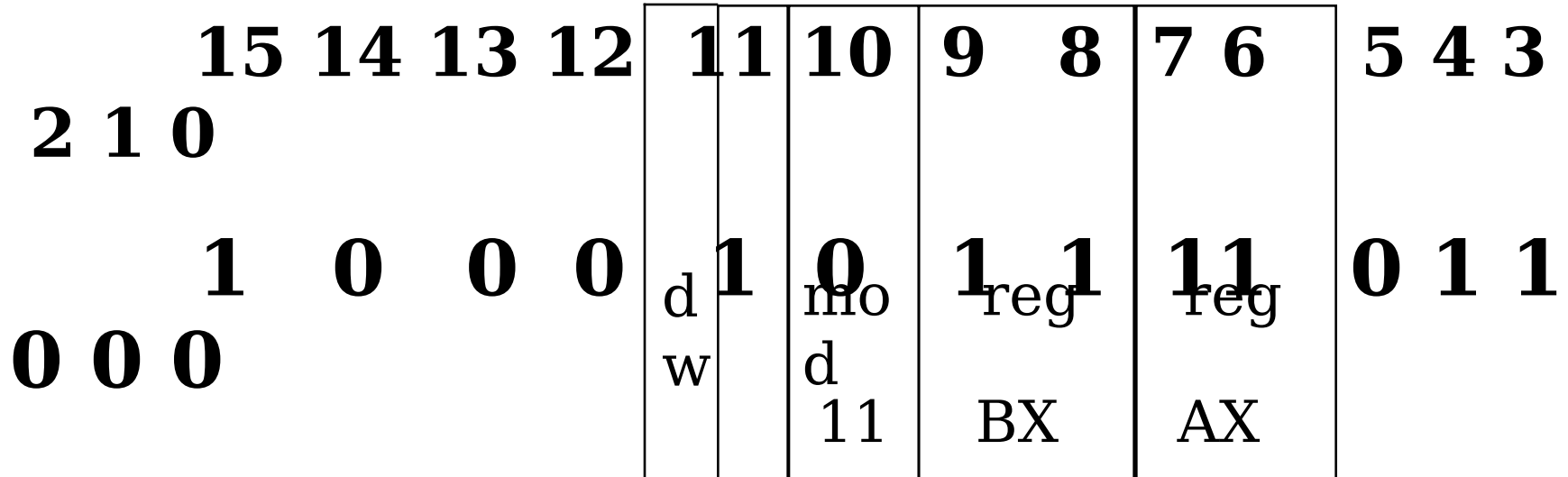
Direct
Transfer



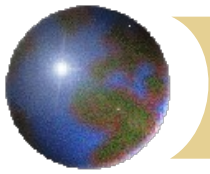
EXAMPLE

MOV BX, AX

8B D8



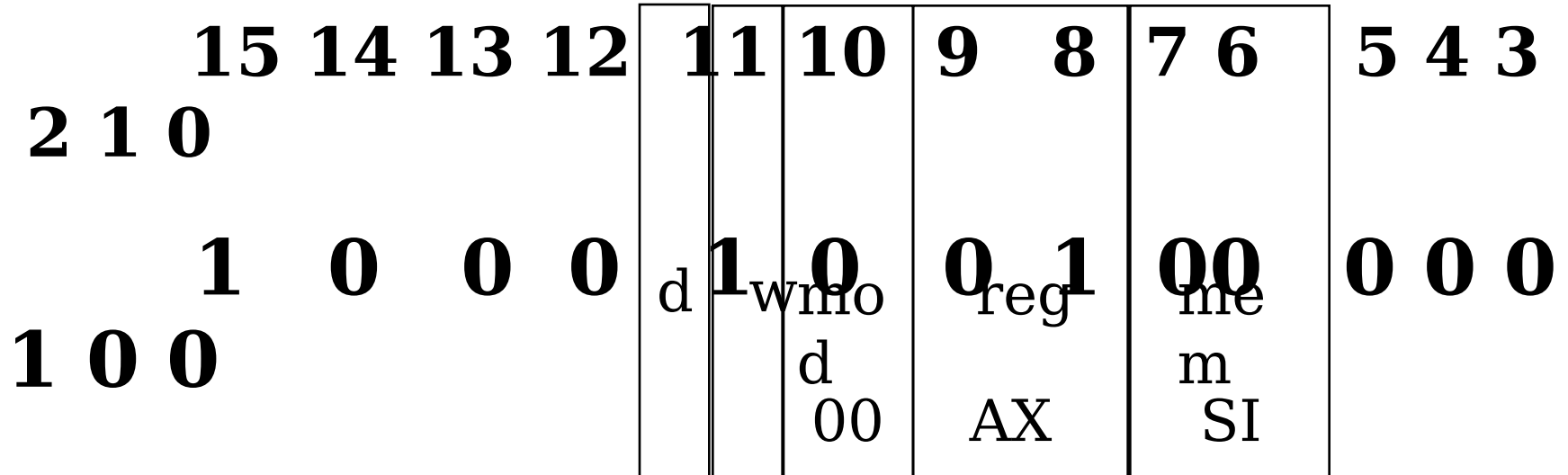
See chart on page A-15



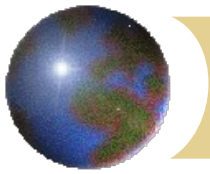
EXAMPLE

MOV M, AX

89 04



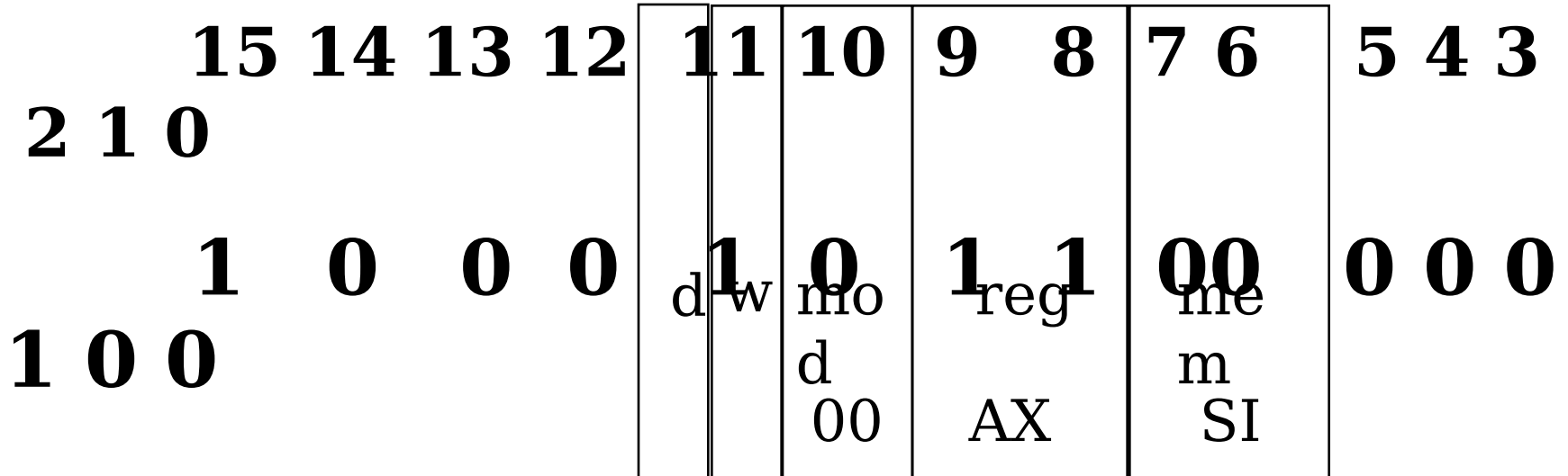
See chart on page A-15



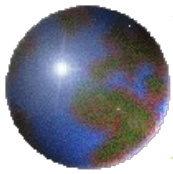
EXAMPLE

MOV AX, M

8B 04



See chart on page A-15



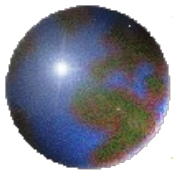
EXAMPLE

MVI BX, data

BB data

7	6	5	3	2	1	15	14	13	11	10	9	7	6	5	3	2	1
4			0			12			8			4			0		
1	0	1	1	0	1	X	X	X	X	X	X	X	X	X	X	X	X
1			W						X		Data	X					
					reg												
					BX												

See chart on page A-15



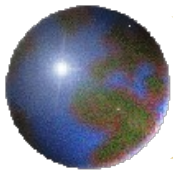
EXAMPLE

INPUT WORD

E5

7	6	5	4	3	2	1	0
1	1	1	0	0	1	0	1
						W	

See chart on page A-16



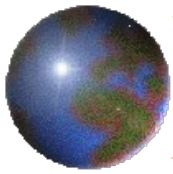
EXAMPLE

OUTPUT WORD

E7

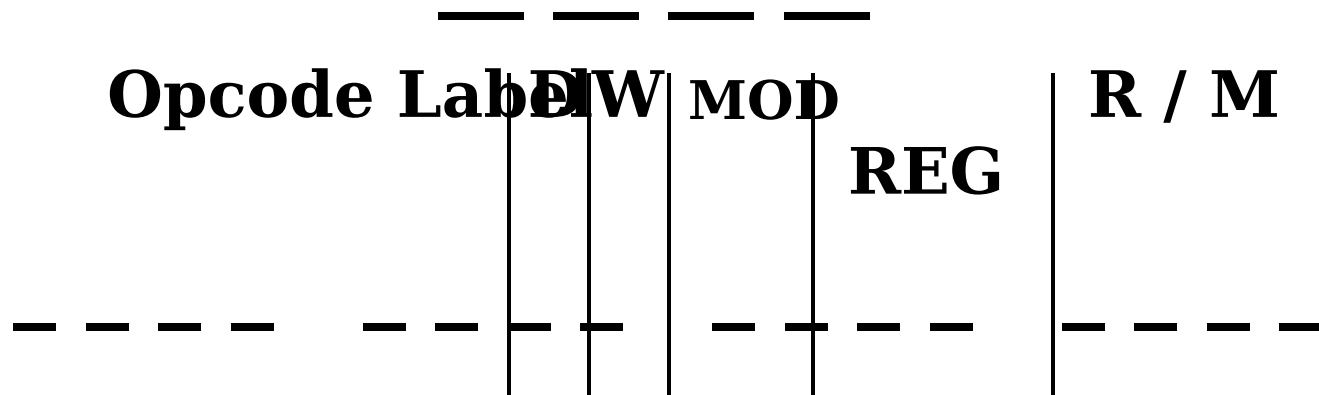
7	6	5	4	3	2	1	0
1	1	1	0	0	1	1	1
						W	

See chart on page A-16



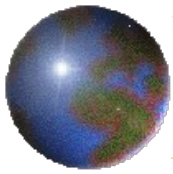
PROBLEM

ADD AX, BX



What is the OP CODE to add the contents of BX to the contents of AX?

See chart on page A-17



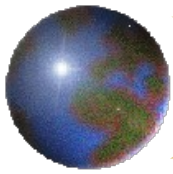
ANSWER

ADD AX, BX

03 C3

Opcode						DW		MOD		REG				R / M			
Label																	
0	0	0	0	0	0	1	1	1	1	0	0			0	0	1	1
										AX				BX			

See chart on page A-17



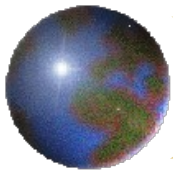
PROBLEM

SUB AX, BX



What is the OP-CODE to subtract the contents of BX from the contents of AX?

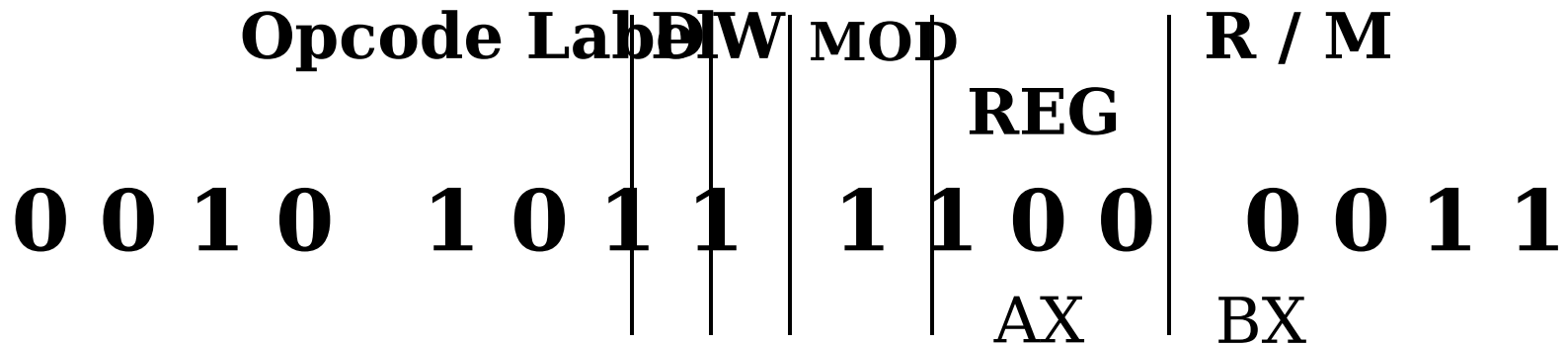
See chart on page A-17



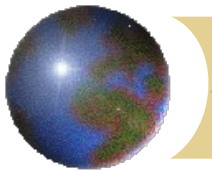
ANSWER

SUB AX, BX

2 B C3



See chart on page A-17



ENABLING LEARNING *OBJECTIVE*

ACTION:

Generate the correct op codes for a list of instructions.

CONDITIONS:

Given a classroom environment, a NIDA 500 trainer, the 8086 instruction set and official OMEMS reference material.

STANDARDS:

Correctly answer 7 of 10 questions on the construction of operation codes.



INSTRUCTIONS

Move 11 into the AL register.

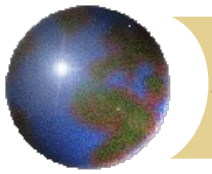
Move 22 into the AH register.

Decrement the contents of the DX register.

AND the contents of the BX to the contents of the DI.

Move the contents of the CX register to the AX.

Subtract immediate number 44 from the



ENABLING LEARNING OBJECTIVE

ACTION:

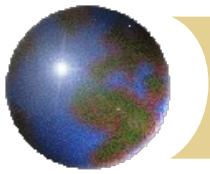
Generate the correct op codes for the sample programs.

CONDITIONS:

Given a classroom environment, a NIDA 500 trainer, the 8086 instruction set and official OMEMS reference material.

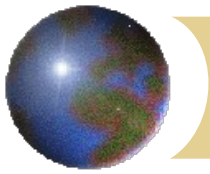
STANDARDS:

Correctly answer 7 of 10 questions on the construction of operation codes.



PRACTICAL EXERCISE I

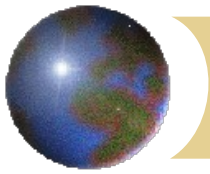
Write a program that loads BCD number 19 into one register and BCD number 17 in another register and adds them together and displays the answer on the output port LEDs.



PRACTICAL EXERCISE II

Write a program that counts to ten and then resets itself to start the count over again.

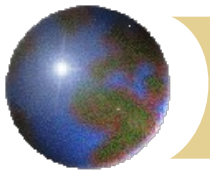
(The program should have provisions to allow the process to run slow enough to see each count on the output port LEDs)



PRACTICAL EXERCISE III

Write a program that will light the output LEDs in and alternating fashion by using the numbers 5555 and AAAA.

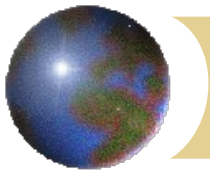
(The program needs to be slow enough to see the lights alternate)



PRACTICAL EXERCISE IV

will make one light shift from one end of the Write a program that output port to the other and back again in a continuous fashion.

(The program needs to be slow enough to see the light shift)



PRACTICAL EXERCISE V

Write a program that will make two lights shift from opposite ends of the output port to the other and back again in a continuous fashion.

(The program needs to be slow enough to see the lights shift)